‘Fall 2022 5’ Group Project (BANA 6620)

Group members: Jun Seok Song(Group Leader), Supraja Palle, Praneeth Reddy Kothwal, Gyanendra Pal, Christian Sabo
Subject: An analysis of influencing factors on housing price: focusing on housing price in Boston
Date: October 27th, 2022

Executive Summary

Analytical Overview

It is very important to analyze what influencing factors can affect housing price and to what extent those factors can affect housing price. In this project, we analyzed the influencing factors on housing price using the 1978 US Census Service dataset for the Boston area (https://www.cs.toronto.edu/~delve/data/boston/bostonDetail.html?ref=morioh.com&utm_source=morioh.com).

To analyze this research subject and dataset, we utilized web scrapping, descriptive analysis, correlation analysis, data visualization, and multiple linear regression with Python.

Major Results

Multiple regression analysis was conducted by setting 12 variables as independent variables and MEDV (housing price) variable as a dependent variable. The following regression equation was derived using the regression coefficients and the intercept.

MEDV = 38.753 + (-0.125)*CRIM + (0.033)*ZN + (0.001)*INDUS + (2.971)*CHAS + (-15.271)*NOX + (3.942)*RM + (-0.004)*AGE + (-1.360)*DIS + (0.306)*RAD + (-0.014)*TAX + (-0.956)* PTRATIO + (-0.543)* LSTAT

According to the results of R^2 (0.6581) and adjusted R^2 (0.6115) derived using testing data and prediction data, it is judged that the multiple regression model using 13 variables has a significant level of fitness and predictive power

Conclusion

According to the regression equation of multiple regression analysis, If the tract bounds the Charles River, the NOX is low, the number of rooms is many, the highway is more accessible, and the distance from the employment center is close, the housing price increases. In addition, the low crime rate, the high residential area ratio, the high non-retail business area ratio, the low age of housing, the low tax, the low student-teacher ratio, and the low lower status ratio also have some effect on the increase in housing prices.

< Description of 13 variables >

CRIM	per capita crime rate by town
ZN	proportion of residential land zoned for lots over 25,000 sq.ft.
INDUS	proportion of non-retail business acres per town
CHAS	Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
NOX	nitric oxides concentration (parts per 10 million)
RM	average number of rooms per dwelling
AGE	proportion of owner-occupied units built prior to 1940
DIS	weighted distances to five Boston employment centres
RAD	index of accessibility to radial highways
TAX	full-value property-tax rate per $10,000
PTRATIO	pupil-teacher ratio by town
LSTAT	% lower status of the population
MEDV	Median value of owner-occupied homes in $1000's

DOCUMENTATION PAGES

Importance of the project

Housing price is the most important and essential item of a household's economic expenditure. In reality, various factors such as the structure, environment, and tax can affect housing price. Therefore, it is very important to analyze what influencing factors can affect housing price and to what extent those factors can affect housing price. In this project, we analyzed the influencing factors on housing price using the 1978 US Census Service dataset for the Boston area.

Dataset description

The dataset for this project was obtained from the Boston Housing Dataset webpage(https://www.cs.toronto.edu/~delve/data/boston/bostonDetail.html?ref=morioh.com&utm_source=morioh.com).

This dataset contains 506 cases and 14 variables such as the median value of owner-occupied homes (housing price), per capita crime rate, the average number of rooms, etc.

The Boston Housing Dataset

A Dataset derived from information collected by the U.S. Census Service concerning housing in the area of Boston Mass.

Delve

This dataset contains information collected by the U.S Census Service concerning housing in the area of Boston Mass. It was obtained from the StatLib archive (http://lib.stat.cmu.edu/datasets/boston), and has been used extensively throughout the literature to benchmark algorithms. However, these comparisons were primarily done outside of **Delve** and are thus somewhat suspect. The dataset is small in size with only 506 cases.

The data was originally published by Harrison, D. and Rubinfeld, D.L. `*Hedonic prices and the demand for clean air*', J. Environ. Economics & Management, vol.5, 81-102, 1978.

Dataset Naming

The name for this dataset is simply **boston**. It has two prototasks: **nox**, in which the nitrous oxide level is to be predicted; and **price**, in which the median value of a home is to be predicted

Miscellaneous Details

- **Origin**
  The origin of the boston housing data is **Natural**.
- **Usage**
  This dataset may be used for **Assessment**.
- **Number of Cases**
  The dataset contains a total of **506** cases.
- **Order**
  The order of the cases is **mysterious**.
- **Variables**
  There are **14** attributes in each case of the dataset. They are:

The Boston house-price data of Harrison, D. and Rubinfeld, D.L. 'Hedonic prices and the demand for clean air', J. Environ. Economics & Management, vol.5, 81-102, 1978. Used in Belsley, Kuh & Welsch, 'Regression diagnostics ...', Wiley, 1980. N.B. Various transformations are used in the table on pages 244-261 of the latter.

Variables in order:

CRIM per capita crime rate by town
ZN proportion of residential land zoned for lots over 25,000 sq.ft.
INDUS proportion of non-retail business acres per town
CHAS Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
NOX nitric oxides concentration (parts per 10 million)
RM average number of rooms per dwelling
AGE proportion of owner-occupied units built prior to 1940
DIS weighted distances to five Boston employment centres
RAD index of accessibility to radial highways
TAX full-value property-tax rate per $10,000
PTRATIO pupil-teacher ratio by town
B 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
LSTAT % lower status of the population
MEDV Median value of owner-occupied homes in $1000's

```
0.00632  18.00   2.310  0  0.5380  6.5750  65.20  4.0900   1  296.0  15.30
 396.90   4.98  24.00
0.02731   0.00   7.070  0  0.4690  6.4210  78.90  4.9671   2  242.0  17.80
 396.90   9.14  21.60
0.02729   0.00   7.070  0  0.4690  7.1850  61.10  4.9671   2  242.0  17.80
 392.83   4.03  34.70
0.03237   0.00   2.180  0  0.4580  6.9980  45.80  6.0622   3  222.0  18.70
 394.63   2.94  33.40
0.06905   0.00   2.180  0  0.4580  7.1470  54.20  6.0622   3  222.0  18.70
 396.90   5.33  36.20
0.02985   0.00   2.180  0  0.4580  6.4300  58.70  6.0622   3  222.0  18.70
 394.12   5.21  28.70
0.08829  12.50   7.870  0  0.5240  6.0120  66.60  5.5605   5  311.0  15.20
 395.60  12.43  22.90
0.14455  12.50   7.870  0  0.5240  6.1720  96.10  5.9505   5  311.0  15.20
 396.90  19.15  27.10
0.21124  12.50   7.870  0  0.5240  5.6310 100.00  6.0821   5  311.0  15.20
 386.63  29.93  16.50
```

Methodology for analysis

To analyze this research topic and dataset, we utilized web scrapping, descriptive analysis, data visualization, and multiple linear regression with Python.

Web scraping is a technique that uses a dataset written on a web page for analysis. Boston housing price dataset is not in the form of .csv file, but a dataset on a web page that requires web scraping to utilize it. Moreover, since the dataset has 14 variables including housing price (MEDV) which is a continuous variable, it is appropriate to use multiple linear regression to analyze the influencing factors on housing price. In addition, descriptive analysis of various attributes of the dataset and data visualization are used together.

Web scrapping

```
In [1]: import urllib3
   ...: import re
   ...: address = 'http://lib.stat.cmu.edu/datasets/boston'
   ...: http = urllib3.PoolManager()
   ...: response = http.request('GET', address)
   ...: webContent = str(response.data)
   ...:
   ...:
   ...: tableContentStart = webContent.find('0.00632')
   ...: tableContent = webContent[tableContentStart:]
   ...: tableData = re.findall('\d+\.\d+|\d+',tableContent)
```

```
In [2]: tableData[0:14]
Out[2]:
['0.00632',
 '18.00',
 '2.310',
 '0',
 '0.5380',
 '6.5750',
 '65.20',
 '4.0900',
 '1',
 '296.0',
 '15.30',
 '396.90',
 '4.98',
 '24.00']

In [3]: len(tableData)
Out[3]: 7084
```

```
In [3]: CRIM = tableData[0::14]
   ...: ZN = tableData[1::14]
   ...: INDUS = tableData[2::14]
   ...: CHAS = tableData[3::14]
   ...: NOX = tableData[4::14]
   ...: RM = tableData[5::14]
   ...: AGE = tableData[6::14]
   ...: DIS = tableData[7::14]
   ...: RAD = tableData[8::14]
   ...: TAX = tableData[9::14]
   ...: PTRATIO = tableData[10::14]
   ...: B = tableData[11::14]
   ...: LSTAT = tableData[12::14]
   ...: MEDV = tableData[13::14]
```

Making a data frame

```
In [4]: import pandas as pd
   ...: import numpy as np
   ...:
   ...:
   ...: df = {'CRIM' : pd.Series(CRIM),
   ...: 'ZN' : pd.Series(ZN),
   ...: 'INDUS' : pd.Series(INDUS),
   ...: 'CHAS' : pd.Series(CHAS),
   ...: 'NOX' : pd.Series(NOX),
   ...: 'RM' : pd.Series(RM),
   ...: 'AGE' : pd.Series(AGE),
   ...: 'DIS' : pd.Series(DIS),
   ...: 'RAD' : pd.Series(RAD),
   ...: 'TAX' : pd.Series(TAX),
   ...: 'PTRATIO' : pd.Series(PTRATIO),
   ...: 'B' : pd.Series(B),
   ...: 'LSTAT' : pd.Series(LSTAT),
   ...: 'MEDV' : pd.Series(MEDV)
   ...: }
   ...: tableData1 = pd.DataFrame(df)
   ...: print(tableData1)
```

```
        CRIM     ZN   INDUS CHAS     NOX  ...    TAX PTRATIO       B LSTAT   MEDV
0    0.00632  18.00   2.310    0  0.5380  ...  296.0   15.30  396.90  4.98  24.00
1    0.02731   0.00   7.070    0  0.4690  ...  242.0   17.80  396.90  9.14  21.60
2    0.02729   0.00   7.070    0  0.4690  ...  242.0   17.80  392.83  4.03  34.70
3    0.03237   0.00   2.180    0  0.4580  ...  222.0   18.70  394.63  2.94  33.40
4    0.06905   0.00   2.180    0  0.4580  ...  222.0   18.70  396.90  5.33  36.20
..       ...    ...     ...  ...     ...  ...    ...     ...     ...   ...    ...
501  0.06263   0.00  11.930    0  0.5730  ...  273.0   21.00  391.99  9.67  22.40
502  0.04527   0.00  11.930    0  0.5730  ...  273.0   21.00  396.90  9.08  20.60
503  0.06076   0.00  11.930    0  0.5730  ...  273.0   21.00  396.90  5.64  23.90
504  0.10959   0.00  11.930    0  0.5730  ...  273.0   21.00  393.45  6.48  22.00
505  0.04741   0.00  11.930    0  0.5730  ...  273.0   21.00  396.90  7.88  11.90

[506 rows x 14 columns]
```

```
In [5]: tableData1.dtypes
Out[5]:
CRIM       object
ZN         object
INDUS      object
CHAS       object
NOX        object
RM         object
AGE        object
DIS        object
RAD        object
TAX        object
PTRATIO    object
B          object
LSTAT      object
MEDV       object
dtype: object
```

Changing data types

The data types of variables were changed to perform descriptive statistical analysis, correlation analysis, and regression analysis for 14 variables (columns). In particular, the dummy variables CHAS and the index variable RAD were set to the integer data type, and the other 12 variables were set to the float data type because they were continuous variables.

```
In [6]: tableData11 = tableData1.astype(float)
   ...: tableData11['RAD'] = tableData11['RAD'].astype(int)
   ...: tableData11['CHAS'] = tableData11['CHAS'].astype(int)
   ...: tableData11.dtypes
Out[6]:
CRIM       float64
ZN         float64
INDUS      float64
CHAS         int32
NOX        float64
RM         float64
AGE        float64
DIS        float64
RAD          int32
TAX        float64
PTRATIO    float64
B          float64
LSTAT      float64
MEDV       float64
dtype: object
```

Deleting the column 'B'

The column 'B' of the dataset is very sensitive to be analyzed because it is a racial variable, so we deleted it from the dataset.

```
In [7]: del tableData11['B']

In [8]: for colName in tableData11:
   ...:     print(colName)
CRIM
ZN
INDUS
CHAS
NOX
RM
AGE
DIS
RAD
TAX
PTRATIO
LSTAT
MEDV
```

Checking out missing values

```
In [9]: tableData11.isnull().sum()
Out[9]:
CRIM       0
ZN         0
INDUS      0
CHAS       0
NOX        0
RM         0
AGE        0
DIS        0
RAD        0
TAX        0
PTRATIO    0
LSTAT      0
MEDV       0
dtype: int64
```

Descriptive statistics

```
In [10]: tableData11['CRIM'].describe(include='all')
Out[10]:
count    506.000000
mean       3.613524
std        8.601545
min        0.006320
25%        0.082045
50%        0.256510
75%        3.677083
max       88.976200
Name: CRIM, dtype: float64

In [11]: tableData11['ZN'].describe(include='all')
Out[11]:
count    506.000000
mean      11.363636
std       23.322453
min        0.000000
25%        0.000000
50%        0.000000
75%       12.500000
max      100.000000
Name: ZN, dtype: float64

In [12]: tableData11['INDUS'].describe(include='all')
Out[12]:
count    506.000000
mean      11.136779
std        6.860353
min        0.460000
25%        5.190000
50%        9.690000
75%       18.100000
max       27.740000
Name: INDUS, dtype: float64

In [13]: tableData11['NOX'].describe(include='all')
Out[13]:
count    506.000000
mean       0.554695
std        0.115878
min        0.385000
25%        0.449000
50%        0.538000
75%        0.624000
max        0.871000
Name: NOX, dtype: float64
```

```
In [14]: tableData11['RM'].describe(include='all')
Out[14]:
count    506.000000
mean       6.284634
std        0.702617
min        3.561000
25%        5.885500
50%        6.208500
75%        6.623500
max        8.780000
Name: RM, dtype: float64

In [15]: tableData11['AGE'].describe(include='all')
Out[15]:
count    506.000000
mean      68.574901
std       28.148861
min        2.900000
25%       45.025000
50%       77.500000
75%       94.075000
max      100.000000
Name: AGE, dtype: float64
```

```
In [16]: tableData11['DIS'].describe(include='all')
Out[16]:
count    506.000000
mean       3.795043
std        2.105710
min        1.129600
25%        2.100175
50%        3.207450
75%        5.188425
max       12.126500
Name: DIS, dtype: float64

In [17]: tableData11['TAX'].describe(include='all')
Out[17]:
count    506.000000
mean     408.237154
std      168.537116
min      187.000000
25%      279.000000
50%      330.000000
75%      666.000000
max      711.000000
Name: TAX, dtype: float64
```

```
In [18]: tableData11['PTRATIO'].describe(include='all')
Out[18]:
count    506.000000
mean      18.455534
std        2.164946
min       12.600000
25%       17.400000
50%       19.050000
75%       20.200000
max       22.000000
Name: PTRATIO, dtype: float64

In [19]: tableData11['LSTAT'].describe(include='all')
Out[19]:
count    506.000000
mean      12.653063
std        7.141062
min        1.730000
25%        6.950000
50%       11.360000
75%       16.955000
max       37.970000
Name: LSTAT, dtype: float64
```

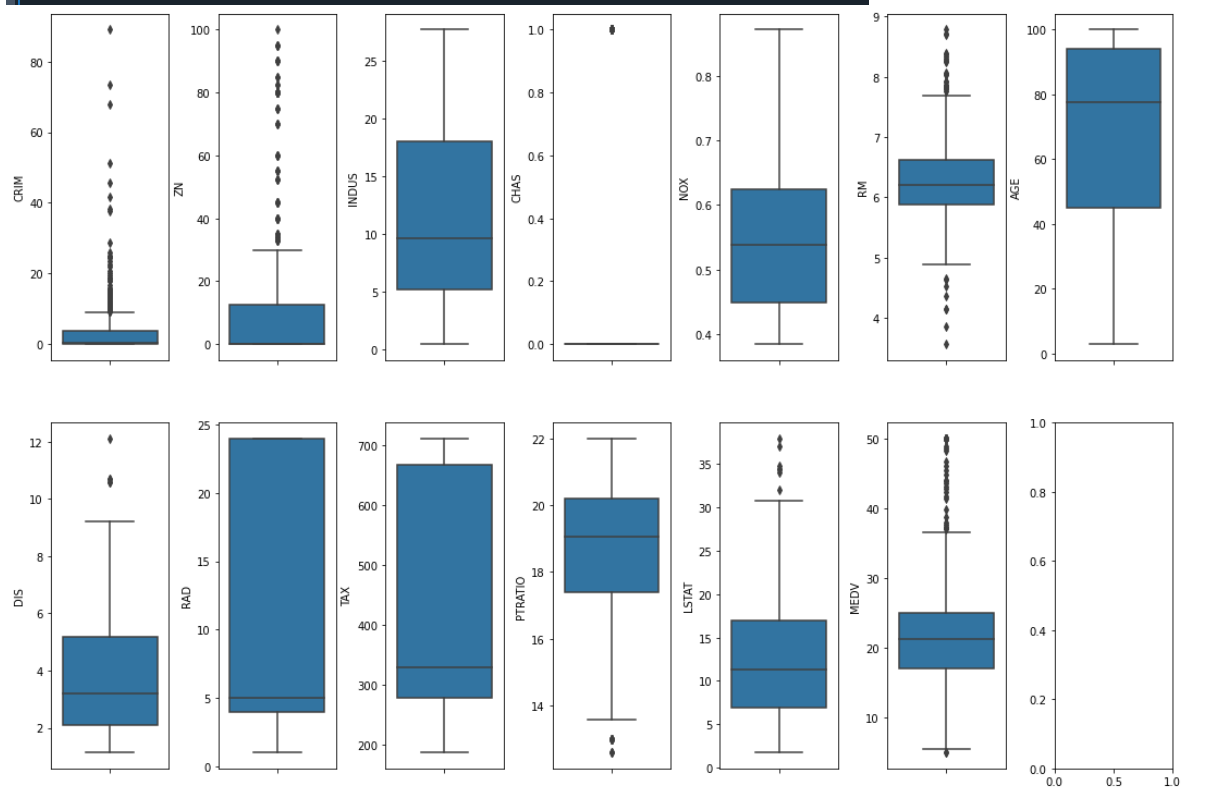
```
In [20]: tableData11['MEDV'].describe(include='all')
Out[20]:
count    506.000000
mean      22.532806
std        9.197104
min        5.000000
25%       17.025000
50%       21.200000
75%       25.000000
max       50.000000
Name: MEDV, dtype: float64

In [21]: tableData11['CHAS'].unique()
Out[21]: array([0, 1])

In [22]: tableData11['RAD'].unique()
Out[22]: array([ 1,  2,  3,  5,  4,  8,  6,  7, 24])
```

Box plots

```
In [23]: import matplotlib.pyplot as plt
    ...: import seaborn as sns
    ...:
    ...: fig, axs = plt.subplots(ncols=7, nrows=2, figsize=(15, 10))
    ...: index = 0
    ...: axs = axs.flatten()
    ...: for i,j in tableData11.items():
    ...:     sns.boxplot(y=i, data=tableData11, ax=axs[index])
    ...:     index += 1
    ...: plt.tight_layout(pad=0.4, w_pad=0.5, h_pad=5.0)
```

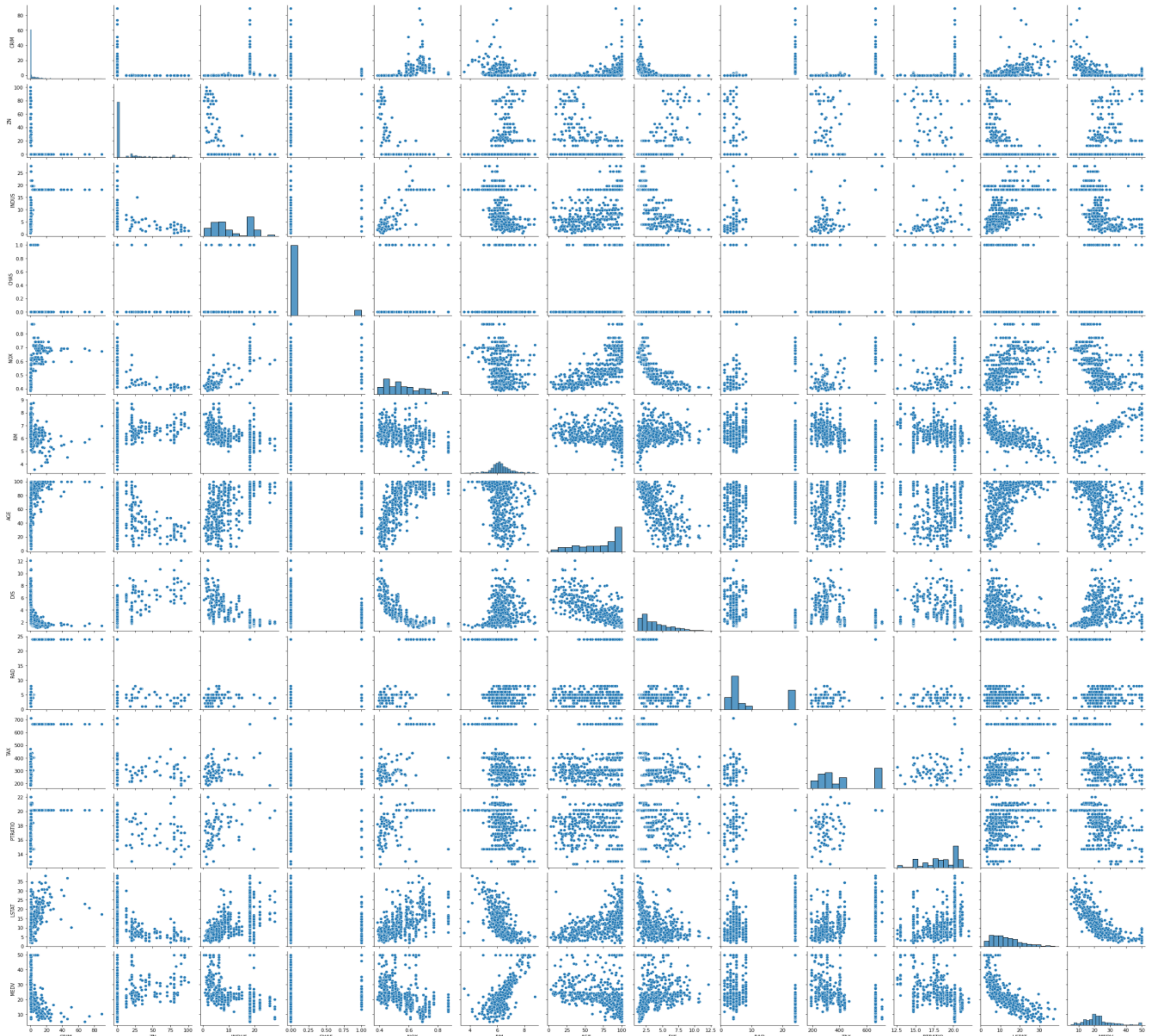


→ Outliers are observed in some variables. However, since there are no missing values in the dataset and the outlier may have its own meaning, subsequent analysis was conducted without artificially deleting the outlier.

Pair plot (Scatter plots + Histograms)

Histograms and scatter plots were examined to find out the distributions of 14 variables and the relationships between the two variables. As a result of examining the scatter plot, linear relationships between some variables such as INDUS, RM, NOX, and LSTAT, and MEDV variable were observed.

```
In [24]: sns.pairplot(tableData11)
Out[24]: <seaborn.axisgrid.PairGrid at 0x275382242e0>
```



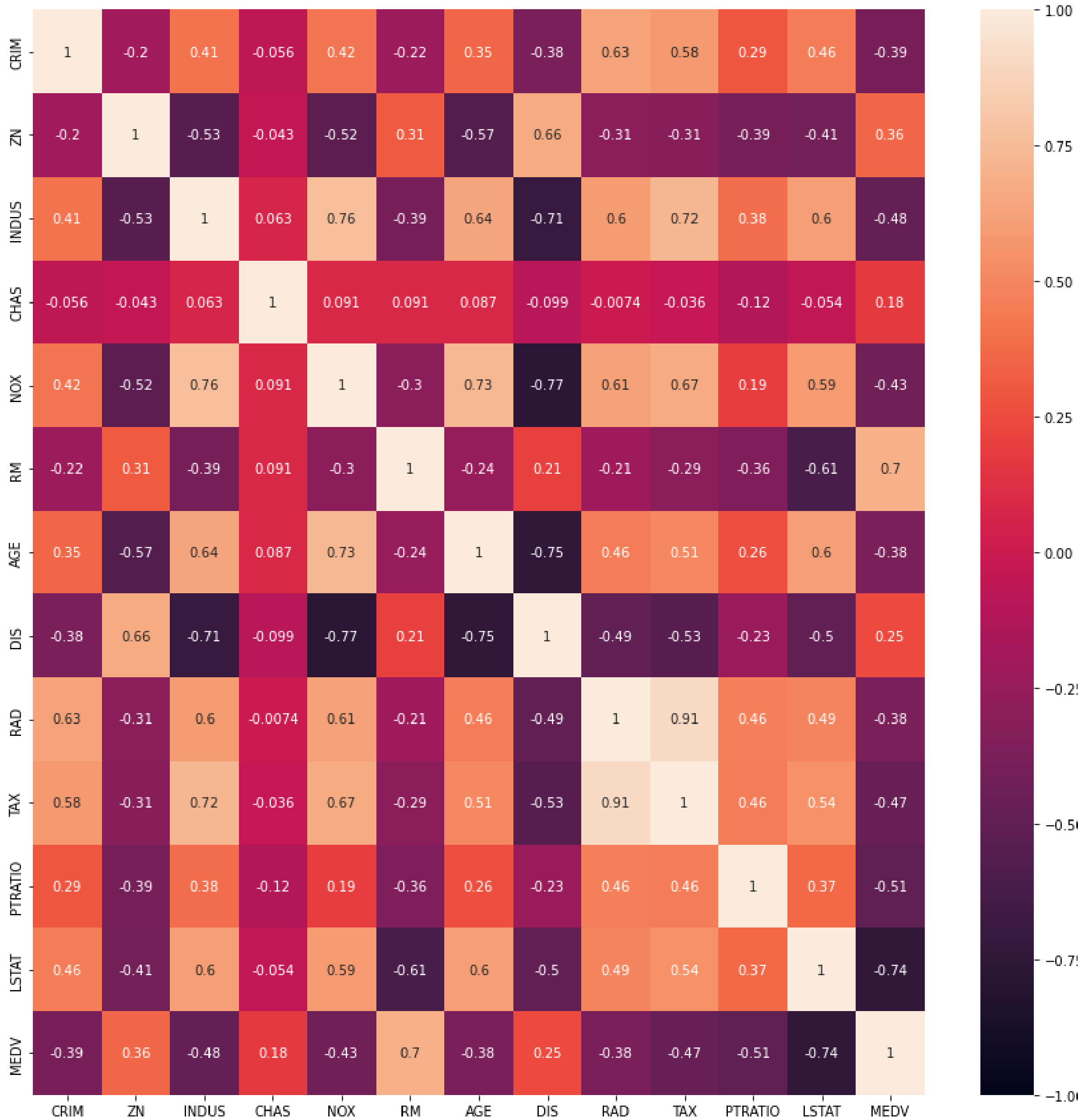
Correlation analysis

The correlations between each of the two variables were analyzed through the correlation coefficient and the heat map. It is shown that there are quite significant correlations between some variables such as INDUS, NOX, RM, TAX, PTRATIO and LSTAT, and the MEDV variable.

```
In [25]: print(tableData11.corr())
             CRIM        ZN     INDUS  ...   PTRATIO     LSTAT      MEDV
CRIM     1.000000 -0.200469  0.406583  ...  0.289946  0.455621 -0.388305
ZN      -0.200469  1.000000 -0.533828  ... -0.391679 -0.412995  0.360445
INDUS    0.406583 -0.533828  1.000000  ...  0.383248  0.603800 -0.483725
CHAS    -0.055892 -0.042697  0.062938  ... -0.121515 -0.053929  0.175260
NOX      0.420972 -0.516604  0.763651  ...  0.188933  0.590879 -0.427321
RM      -0.219247  0.311991 -0.391676  ... -0.355501 -0.613808  0.695360
AGE      0.352734 -0.569537  0.644779  ...  0.261515  0.602339 -0.376955
DIS     -0.379670  0.664408 -0.708027  ... -0.232471 -0.496996  0.249929
RAD      0.625505 -0.311948  0.595129  ...  0.464741  0.488676 -0.381626
TAX      0.582764 -0.314563  0.720760  ...  0.460853  0.543993 -0.468536
PTRATIO  0.289946 -0.391679  0.383248  ...  1.000000  0.374044 -0.507787
LSTAT    0.455621 -0.412995  0.603800  ...  0.374044  1.000000 -0.737663
MEDV    -0.388305  0.360445 -0.483725  ... -0.507787 -0.737663  1.000000

[13 rows x 13 columns]
```

```
In [26]: plt.figure(figsize = (15,15))
    ...: sns.heatmap(tableData11.corr(), vmin=-1, vmax=1, annot=True)
Out[26]: <AxesSubplot:>
```



Dividing data into training data and testing data

Prior to multiple regression analysis, 405 (80%) of the 506 observations of the data were randomly assigned as training data and the remaining 101 (20%) as testing data to prevent overfitting of the data and verify the accuracy of the prediction.

```
In [27]: np.random.seed(10)
    ...: numberRows = len(tableData11)
    ...: randomlyShuffledRows = np.random.permutation(numberRows)
    ...: randomlyShuffledRows
    ...:
    ...: trainingRows = randomlyShuffledRows[0:405]
    ...: testRows = randomlyShuffledRows[405:]
    ...:
    ...:
    ...: xTrainingData = tableData11.iloc[trainingRows,0:-1]
    ...: yTrainingData = tableData11.iloc[trainingRows,-1]
    ...:
    ...: xTestData = tableData11.iloc[testRows,0:-1]
    ...: yTestData = tableData11.iloc[testRows,-1]
```

Multiple linear regression

```
In [28]: from sklearn import linear_model
    ...:
    ...: reg = linear_model.LinearRegression()
    ...: reg.fit(xTrainingData,yTrainingData)
Out[28]: LinearRegression()

In [29]: print(reg.coef_)
[-1.24918124e-01  3.32128299e-02  1.33812411e-03  2.97105114e+00
 -1.52709220e+01  3.94205181e+00 -4.19683876e-03 -1.35975824e+00
  3.06334641e-01 -1.38625832e-02 -9.55873269e-01 -5.43480934e-01]

In [30]: print(reg.intercept_)
38.75264682038632
```

→ Multiple regression analysis was conducted by setting 12 variables as independent variables and MEDV variable as a dependent variable. The following regression equation was derived using the regression coefficients and the intercept.

MEDV = 38.753 + (-0.125)*CRIM + (0.033)*ZN + (0.001)*INDUS + (2.971)*CHAS + (-15.271)*NOX + (3.942)*RM + (-0.004)*AGE + (-1.360)*DIS + (0.306)*RAD + (-0.014)*TAX + (-0.956)* PTRATIO + (-0.543)* LSTAT

Evaluation of the multiple linear regression model

< Mean Squared Error (MSE) >

```
In [31]: yPredictions = reg.predict(xTestData)
    ...: errors = (yPredictions-yTestData)
    ...:
    ...: from sklearn.metrics import mean_squared_error
    ...: mse = mean_squared_error(yTestData,yPredictions)
    ...:
    ...: print(mse)
25.76254687262308
```

< R^2 >

```
In [32]: from sklearn.metrics import r2_score
    ...: r2 = r2_score(yTestData,yPredictions)
    ...:
    ...: print(r2)
0.6580900612207468
```

< Adjusted R^2 >

```
In [33]: adj_r2 = 1 - (1-r2)*(len(xTestData)-1)/(len(xTestData)-len(xTestData.columns)-1)
    ...: print(adj_r2)
0.6114659786599395
```

→ According to the results of MSE, R^2, and adjusted R^2 derived using testing data and prediction data, it is judged that the multiple regression model using 13 variables has a significant level of fitness and predictive power